Universidad Nacional de Chimborazo
NOVASINERGIA, 2018, Vol. 1, No. 1, Junio, (33-40)
ISSN: 2631-2654

Efecto de los coeficientes de aceleración de PSO en el desempeño de una Red Neuronal Artificial aplicada a la Estimación de Costos

Effect of the PSO acceleration coefficients on the performance of an Artificial Neural Network applied to the Cost Estimation

Elba Bodero Poveda[1], Guillermo Leguizamón[2*]
[1]*Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina, 1900*
[2]*Departamento de Informática, Universidad Nacional de San Luis, San Luis, Argentina, 5700*
legui@unsl.edu.ar

Resumen: La metaheurística poblacional *Particle Swarm Optimization* (PSO) desde su aparición ha demostrado ser eficiente en la solución de problemas de optimización, la variación de sus parámetros ha permitido mejorar su eficiencia. El presente trabajo está centrado en realizar un estudio comparativo del efecto de los coeficientes de aceleración, c_1 y c_2, en el desempeño de PSO para resolver un problema de estimación de costos por medio de una Red Neuronal Artificial (ANN) *feedforward* sigmoidal. Se evaluó un rango de valores en los coeficientes de aceleración, los demás parámetros, en este caso factor inercial y el tamaño de enjambre se trabajaron con valores fijos. La validación de la solución se realizó por medio de un conjunto de datos de fabricación de tuberías para transferencia de fluidos utilizada en la industria, proveniente de un caso real, con información relacionada con peso, tipo de soldadura, diámetro y el correspondiente costo. La función objetivo utilizada es el Error Cuadrático Medio (MSE), calculado entre los valores observados y los valores estimados por la ANN. A partir de los resultados se puede observar que valores muy pequeños de c_1 y c_2 obtienen baja exactitud en la estimación de costos de fabricación de tubería, en tanto que la mejor exactitud es lograda por medio de coeficientes de aceleración con valores mayores o iguales a 0.5.

Palabras clave: Coeficientes de Aceleración PSO, Estimación de Costos, Metaheurística Poblacional, *Particle Swarm Optimization*, Red Neuronal Artificial.

Abstract: *The particle metaheuristics Particle Swarm Optimization (PSO) since its appearance has proven to be efficient in solving optimization problems, the variation of its parameters has allowed to improve its efficiency. The present work is focused on performing a comparative study of the effect of the acceleration coefficients c_1 and c_2, on the performance of PSO to solve a problem of cost estimation, through an Artificial Neural Network (ANN) sigmoidal feedforward. A range of values was evaluated in the acceleration coefficients, the other parameters, in this case inertial factor and the swarm size were worked with fixed values. The validation of the solution was carried out by means of a pipeline data set for fluid transfer used in the industry, coming from a real case, with information related to weight, welding type, diameter and the corresponding cost. The objective function used is the Mean Square Error (MSE), calculated between the observed values and the values estimated by the ANN. From the results it can be seen that very small values of c_1 and c_2 obtain low accuracy in the estimation of pipe manufacturing costs, while the best accuracy is achieved by means of acceleration coefficients with values greater than or equal to 0.5.*

Keywords: *PSO Acceleration Coefficients, Estimation of Costs, Population Metaheuristics, Particle Swarm Optimization, Artificial Neural Network.*

1 Introducción

Particle Swarm Optimization (PSO) como método estocástico de optimización global inicia con estudios realizados por Kennedy y Eberhart (1995), quienes se fijan como objetivo inicial simular gráficamente el movimiento sincronizado e impredecible de grupos tales como los bancos de peces o las bandadas de aves, intrigados por la capacidad de estos grupos para separarse, reagruparse o encontrar alimento.

Dentro de esta misma línea, con trabajos previos en el ámbito de la biología y de la sociología, que concluyen que el comportamiento, inteligencia y movimiento de estas agrupaciones, entre las cuales se podría incluir con un cierto grado de abstracción a los seres humanos, está relacionado directamente con la capacidad de los individuos para compartir información y aprovecharse de la experiencia semejante acumulada.

Kennedy y Eberhart (1995), modelan dicho comportamiento de forma matemática utilizando expresiones simples que revelan su potencial como método de optimización. En la terminología utilizada en PSO, Kennedy y Eberhart (1995, 2001) introducen el término general partícula o agente para representar a los peces, pájaros, abejas, hormigas o cualquier otro tipo de individuos que exhiban un comportamiento social como grupo, en forma de una colección de agentes que interactúan entre sí.

De acuerdo con los fundamentos teóricos del método, el movimiento de cada una de estas partículas hacia un objetivo común en dos dimensiones está condicionado por dos factores básicos, la memoria autobiográfica de la partícula o nostalgia y la influencia social de todo el enjambre. A nivel computacional, como método de optimización, esta filosofía puede extenderse a un espacio N-dimensional de acuerdo con el problema bajo análisis. La posición instantánea de cada una de las partículas de la población en el espacio N-dimensional representa una solución potencial, siendo N el número de incógnitas del problema original.

Básicamente, el proceso evolutivo se reduce a mover cada partícula dentro del espacio de soluciones con una velocidad que variará de acuerdo a su velocidad actual, a la memoria de la partícula y a la información global que comparte el resto del enjambre, utilizando una función de *fitness* para cuantificar la calidad de cada partícula en función de la posición que ésta ocupe, más allá de la propia naturaleza del método, los esquemas existentes para la implementación son muy diversos.

En la investigación realizada por Carlisle y Dozier (2001) se muestran variantes dependiendo de cómo se actualicen las posiciones de las partículas surgen las versiones síncrona y asíncrona del algoritmo. Adicionalmente, dependiendo de cómo se haga influir la experiencia acumulada por el enjambre sobre el movimiento de cada una de las partículas que lo integran, se puede distinguir entre PSO local y global, como lo indican Eberhart y Shi (2001). Son muy extensas las variantes que los propios autores e investigadores plantean, con el propósito de mejorar el rendimiento del algoritmo original en aplicaciones concretas.

Trasladando la filosofía de PSO al campo de la vida artificial y del cómputo evolutivo, entre las múltiples áreas donde ha sido aplicado con éxito, destacan por su importancia: optimización de funciones y resolución de problemas matemáticos complejos (Laskari *et al*., 2002), (Hu *et al*., 2003), optimización de pronóstico sobre algoritmos clásicos (Barba y Rodríguez, 2015), entrenamiento de redes neuronales (Ismail y Engelbrecht, 2000), (Srinivasan *et al*., 2003), (Eberhart, Hu, 1999), (Wang *et al*., 2004), optimización de sistemas dinámicos (Hu y Eberhart, 2002), (Vesterstrom y Riget, 2002), procesado de señal (Zhao y Zheng, 2004), (Lu y Yan, 2004), gestión, planificación y optimización de recursos en redes de distribución de energía eléctrica (Naka *et al*., 2001), (Koay y Srinivasan, 2003), (Naka *et al*., 2003), (Miranda y Fonseca, 2002), (Gaing, 2003), (Chang y Lu, 2002), gestión de redes de sensores (Veeramachaneni y Osadciw, 2004a), (Veeramachaneni y Osadciw, 2004b), planificación de red en servicios de telecomunicación (Yangyang *et al*., 2004), gestión empresarial (Tasgetiren y Liang, 2003) y teoría de juegos (Franken y Engelbrecht, 2004), entre otros.

En las aplicaciones en el ámbito de la vida artificial se deben respetar cinco principios básicos sobre la inteligencia de grupo (Kennedy y Eberhart, 1995), (Millonas, 1994), estos principios son: proximidad, calidad, diversidad de respuesta, estabilidad y adaptabilidad.

Con base en lo anteriormente expuesto, se plantea un estudio comparativo que muestra los cambios en las constantes de aceleración c_1 y c_2 en la actualización de la posición de las partículas en PSO, con lo que se resolverá un problema de estimación de costos, mediante una Red Neuronal Artificial tipo *feedforward* sigmoidal con aprendizaje PSO. El documento está estructurado de la siguiente manera. La sección 2 se describe la metodología a utilizar, en la sección 3 se presentan los resultados y discusión, y en la sección 4 se muestran las conclusiones.

2 Metodología

En la formulación de PSO se define la velocidad de partícula como el único operador disponible para controlar la evolución de la optimización. Se

considera una población de I partículas donde cada partícula del enjambre se identifica con dos variables de estado inicializadas aleatoriamente dentro del espacio N-dimensional que establece el problema a optimizar: un vector velocidad (1a) y un vector de posición (1b) que corresponde a una solución potencial al problema de optimización:

$$V_i = V_{i1}, V_{i2}, \dots, V_{in}, \quad (1a)$$
$$X_i = X_{i1}, X_{i2}, \dots, X_{in}, \quad (1b)$$
$$X_n \in [X_{n,min}, X_{n,max}]. \quad (1c)$$

Los límites de los parámetros a optimizar (1c) conforman en su conjunto el espacio de búsqueda al cual debe restringirse el movimiento del enjambre.
Adicionalmente cada partícula mantiene en memoria información de la posición espacial asociada con la mejor solución históricamente visitada por ésta (2a) y también conoce la posición de la mejor partícula o solución encontrada por todos sus semejantes (2b). El movimiento del enjambre se realiza en pasos temporales, que se traducen a nivel de algoritmo en iteraciones contiguas.

$$P_i = p_{i1}, p_{i2}, \dots, p_{in} \quad (2a)$$
$$G = g_1, g_2, \dots, g_n \quad (2b)$$

En cada iteración del método, k, cada una de las partículas de la población recorre el espacio de soluciones con una velocidad V_i hacia nuevas posiciones X_i, de acuerdo con su propia experiencia P_i y con la experiencia aportada por el mejor de sus vecinos, G. En las primeras versiones del algoritmo (Kennedy y Eberhart, 1995) ésta formulación se reduce a las ecuaciones mostradas a continuación (3a) y (3b).

$$v_{in}(k+1) = v_{in}(k) + c_1 r_1(k).(p_{in}(k) - x_{in}(k)) + c_2 r_2(k).(g_n(k) - x_{in}(k)), \quad (3a)$$

$$x_{in}(k+1) = x_{in}(k) + v_{in}(k+1).\Delta t. \quad (3b)$$

Entonces, $v_{in}(k)$ y $x_{in}(k)$ representan, la velocidad y posición en la iteración o instante de tiempo k de la partícula i en la dimensión n-ésima del espacio de búsqueda. Los factores c_1 y c_2 son las denominadas constantes de aceleración cognitiva y social, que determinan en qué medida influyen sobre el movimiento de la partícula su propia memoria y la cooperación entre individuos, respectivamente. Los términos $r_1(k)$ y $r_2(k)$ son dos números aleatorios uniformemente distribuidos entre 0 y 1, U [0,1], cuyo objetivo es emular el comportamiento estocástico y un tanto impredecible que exhibe la población del enjambre. Después de calcular la nueva velocidad de la partícula i en la dimensión n, la nueva posición $x_{in}(k+1)$ se actualiza directamente de acuerdo con (3b), donde se asume que la velocidad se aplica durante un cierto período de tiempo Δt, típicamente de valor unitario. El proceso descrito se extiende al espacio N-dimensional, de forma que se van componiendo iterativamente nuevos vectores de posición X_i, utilizando, como en cualquier otro método de cómputo evolutivo una función de fitness para ponderar la calidad de dicha solución parcial, actualizando los vectores P_i y G si se detectan resultados mejores.

El movimiento de los agentes sobre el espacio de soluciones y el rendimiento del algoritmo está condicionado por el grado de contribución de las tres componentes de la velocidad en (3a) y que tienen que ver con el comportamiento social como método de optimización global: hábito o inercia, para considerar la tendencia de la partícula; memoria, nostalgia o autoaprendizaje para incluir la experiencia de la propia partícula, y cooperación, conocimiento social, conocimiento de grupo o información compartida, para reflejar el intercambio de información y el comportamiento social como grupo (Kennedy, 1997).

El procedimiento computacional completo para el algoritmo PSO se puede resumir de la siguiente manera:

Paso 1: Inicialización.

- Establecer tiempo t=0 y número de partículas NP.
- Genera partículas NP al azar $\{X_i^0, i = 1,2, \dots, NP\}$ cuando $X_i^0 = [x_{i1}^0, x_{i2}^0, \dots, x_{in}^0]$.
- Generar las velocidades iniciales para cada partícula aleatoriamente, $\{V_i^0, i = 1,2, \dots, NP\}$ cuando $V_i^0 = [v_{i1}^0, v_{i2}^0, \dots, v_{in}^0]$
- Evaluar cada partícula en el enjambre usando la función objetivo f_i^0 para $i = 1,2, \dots, NP$
- Para cada partícula en el enjambre, establecer $P_i^0 = X_i^0$, cuando $P_i^0 = [p_{i1}^0 = x_{i1}^0, p_{i2}^0 = x_{i2}^0, \dots, p_{in}^0 = x_{in}^0]$ junto con el mejor valor de fitness, f_i^{pb} para $i = 1,2, \dots, NP$
- Encontrar el mejor valor de fitness entre todo el enjambre, de manera que $f_l \min\{f_i^0\}$ para $i = 1,2, \dots, NP$ con sus posiciones correspondientes X_i^0. Establecer el óptimo global a $G^0 = X_i^0$ tal que $G^0 = [g_1 = x_{l,1}, g_2 = x_{l,2}, \dots, g_n = x_{l,n}]$ con su valor de fitness $f^{gb} = f_l$

Paso 2: Actualizar el contador de iteraciones

- t = t +1.

Paso 3: Actualizar el peso inercial.

- $w^t = ((\max\_fes - FES)/\max\_fes) * (w_0 - w_n) + w_n$ donde max_fes, FES, w_0 , y w_n son el número máximo de funciones de evaluación, número de funciones de evaluación, peso inercial inicial y peso inercial final, respectivamente.

Paso 4: Actualizar la velocidad.

. $v_{ij}^{t} = w^{t-1}v_{ij}^{t-1} + c_1 r_1 \left(p_{ij}^{t-1} - x_{ij}^{t-1}\right) + c_2 r_2 \left(g_j^{t-1} - x_{ij}^{t-1}\right)$ donde, c_1 y c_2 son los coeficientes de aceleración; r_1 y r_2 son números aleatorios uniformes entre (0,1).

Paso 5: Actualizar posición

. $x_{ij}^{t} = x_{ij}^{t-1} + v_{ij}^{t}$

Paso 6: Actualizar P_{best}

. Cada partícula se evalúa mediante el uso de permutación para ver si mejora P_{best} , siempre que $f_i^t < f_i^{pb}$ para $i = 1,2,\dots,NP$ entonces el mejor P_{best} se actualiza como: $P_i^t = X_i^t$ y $f_i^{pb} = f_i^t$

Paso 7: Actualizar G_{best}

. Encuentra el valor mínimo P_{best} Es decir, $f_l^t = \min\{f_i^{pb}\}$, $i = 1,2,\dots,NP$; $l \in \{i; i = 1,2,\dots,NP\}$

. Si $f_l^t < f^{gb}$, entonces el G_{best} es actualizado a $G^t = X_l^t$ y $f^{gb} = f_l^t$

Paso 8: Aplicar criterio de parada.

. Si el número de evaluaciones de funciones excede el número máximo de evaluaciones de funciones, para: caso contrario regresa al paso 2.

El modelo de estimación de Redes Neuronales Artificiales tiene una estructura común de tres capas (Freeman y Skapura, 1991); las entradas son peso, tipo de soldadura y diámetro; en la capa oculta se aplica la función de transferencia sigmoidea, y en la capa de salida se obtiene el valor estimado. El resultado de ANN es:

$$\hat{y} = \sum_{j=1}^{Q} v_j h_j, \qquad (4a)$$

$$h_j = f\left[\sum_{i=1}^{K} \omega_{ji} x_i(n)\right]. \qquad (4b)$$

donde $\hat{y}$ es el valor estimado (4a), Q es el número de nodos ocultos, v_j y ω_{ji} son los pesos lineales y no lineales de las conexiones ANN, respectivamente, $x_i$representa *i* ésima variable de entrada. La expresión analítica de la función de transferencia sigmoidea (5) es:

$$f(x) = \frac{1}{1+e^{-x}} \qquad (5)$$

El peso de las conexiones ANN, v y ω se ajustan con el algoritmo de aprendizaje PSO. En la Fig. 1 se ilustra la integración de la ANN con PSO. Se inicializa el enjambre de partículas con un tamaño determinado. Se calcula la función de costo (MSE descrita más adelante), correspondiente a cada peso de las conexiones de la ANN. Cuando el mejor desempeño es alcanzado, en este caso el Error Mínimo, el algoritmo finaliza; caso contrario se realiza una nueva búsqueda de posición para cada partícula.

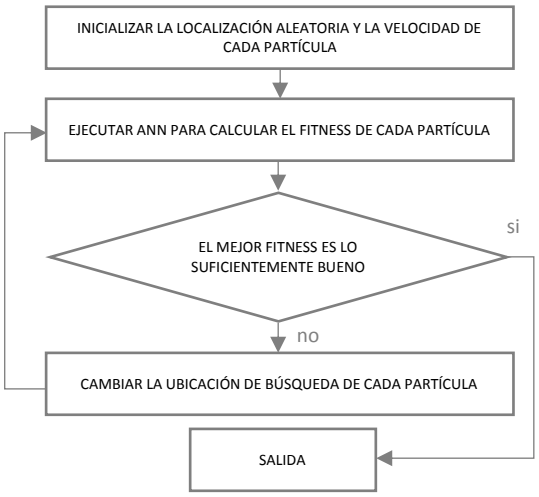

Figura 1: Esquema de funcionamiento de ANN-PSO

Existen diferentes métricas para evaluar el rendimiento de la ANN, en este trabajo se calculan la Raíz Cuadrada del Error Cuadrático Medio (*RMSE*) (6a), Error Cuadrático Medio (*MSE*) (6b) y el Coeficiente de Determinación (R^2) (6c), como se muestra a continuación,

$$RMSE = \sqrt{MSE} \qquad (6a)$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \qquad (6b)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N_S}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{N_S}(x_i - \bar{x}_i)^2} \qquad (6c)$$

Donde, a partir de la muestra de validación, x_i es el i-ésimo valor observado, $\hat{x}_i$ es el i-ésimo valor estimado, $\bar{x}$ es la media del valor observado, y N es el número de muestras.

3 Resultados y Discusión

En la aplicación empírica, una ANN *feedforward* sigmoidal es implementada para evaluar el algoritmo PSO. La ANN es utilizada en este caso para la estimación de costos de construcción industrial.

Para este estudio se utilizaron 2.253 datos de un fabricante real de elementos de tubería para transferencia de fluidos en operaciones de minería a gran escala (Rodríguez y Durán, 2013). La base de datos incluye un conjunto de 6 variables de entrada y 1 de salida. Las variables significativas fueron identificadas por medio de un análisis de correlación

entre las entradas y las salidas, en este caso son, peso, tipo de soldadura y diámetro.

El conjunto de datos se divide en dos partes: un conjunto de datos de entrenamiento (con el 75%) y un conjunto de datos de prueba con el restante 25%.
El éxito del algoritmo radica en su capacidad de ajustar las posiciones de las partículas en un área del espacio de soluciones prometedora, de acuerdo a una función objetivo que se desea minimizar, en este caso el Error Cuadrático Medio (*MSE*).

En la figura 2 se muestran los resultados del desempeño de una ANN (3,5,1), de la función de costo Error Cuadrático Medio (Mean Squared Error, *MSE*), y coeficientes de aceleración, c_1=0.05, c_2=0.05. A partir de la figura, se observa que con más de 200 repeticiones el algoritmo converge en el mínimo. En la figura 3 se han variado los coeficientes a c_1=0.2, c_2=0.2, como resultado se observa que el algoritmo logra el mínimo con alrededor de 400 repeticiones. Mientras que en la figura 4, se ilustran los resultados con los parámetros c_1=0.5, c_2=0.5, la función converge en más de 300 iteraciones. En la figura 5, se presentan los resultados con los parámetros c_1=0.95, c_2=0.05, la función converge en 400 iteraciones, al igual que en la figura 6, la que usa c_1=0.05, c_2=0.95. En la figura 7 se presentan los resultados con los parámetros c_1=0.95, c_2=0.95, la función converge en menos de 400 iteraciones.

Los resultados de las métricas aplicadas para evaluar la exactitud de la estimación por medio de la ANN-PSO se muestran en la tabla 1. El experimento se repitió 3 veces usando la misma configuración. A partir de los resultados se puede observar que valores muy pequeños de c_1 y c_2 (cercanos a cero) obtienen baja exactitud en la estimación de costos de fabricación de tubería, en tanto que la mejor exactitud es lograda por medio de una ANN cuyos coeficientes de aceleración son mayores o iguales a 0.5. Los resultados presentados en la tabla guardan relación con las aseveraciones de Duarte y Quiroga (2010), quienes determinan que los dos coeficientes de aceleración cercanos a cero producirán una búsqueda fina en una región, mientras coeficientes cercanos a uno permitirán a la partícula la posibilidad de sobrepasar al G_{best} y al P_{best} resultando en una búsqueda amplia.

Los valores obtenidos en las métricas *RMSE* y R^2 en el presente experimento, no difieren en mayor medida de los obtenidos por medio de un modelo neuronal más complejo basado en una ANN recurrente usada por Barba y Bodero (2017) para la estimación de costos de tuberías que aplica los mismos datos.
Las condiciones de término comúnmente usadas para finalizar el proceso de optimización son:

- Número máximo de iteraciones: El proceso termina después de alcanzar un número fijo de iteraciones.
- Número de iteraciones sin mejoras: El proceso de iteración termina después que se alcanza un número fijo de iteraciones en la que no se han obtenido alguna mejora en términos de solución.
- Error mínimo de la función objetivo: El error entre el valor obtenido de la función objetivo y el mejor valor de eficacia (*fitness*) es menor que un umbral prefijo esperado.

El mal ajuste de los parámetros puede provocar que el PSO converja a una solución en pocas iteraciones, o a una buena solución en muchas iteraciones. A menudo, PSO puede encontrar una mala solución en pocas iteraciones (conocida como convergencia prematura), o una mala solución en muchas iteraciones.

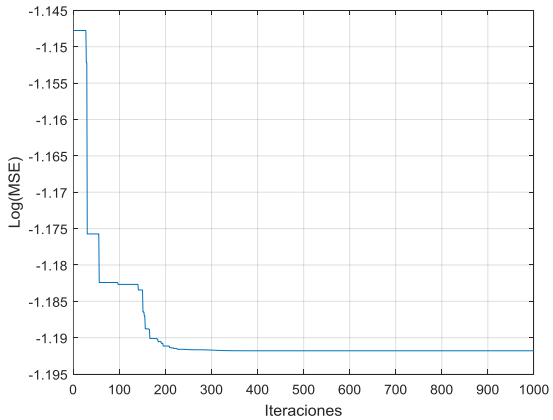

Figura 2: ANN (3,5,1), Fitness *MSE*, c_1=0.05, c_2=0.05

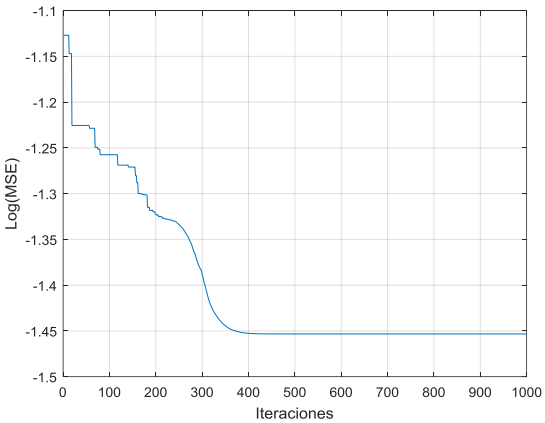

Figura 3: ANN (3,5,1), Fitness *MSE*, c_1=0.2, c_2=0.2

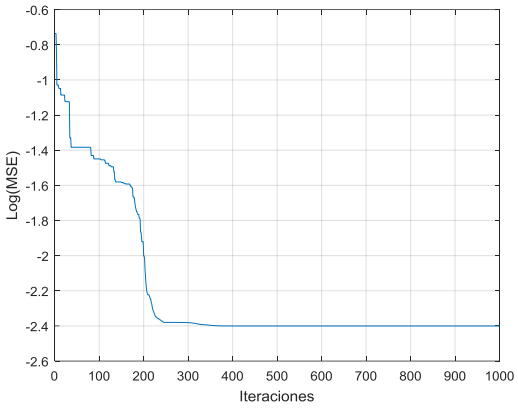

Figura 4: ANN (3,5,1), Fitness *MSE*, c_1=0.5, c_2=0.5

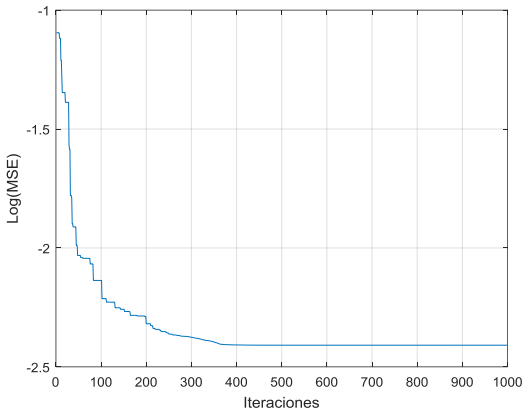

Figura 5: ANN (3,5,1), Fitness *MSE*, c_1=0.05, c_2=0.95

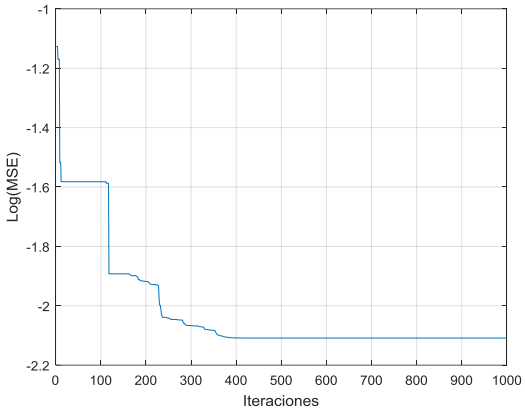

Figura 6: ANN (3,5,1), Fitness *MSE*, c_1=0.95, c_2=0.05

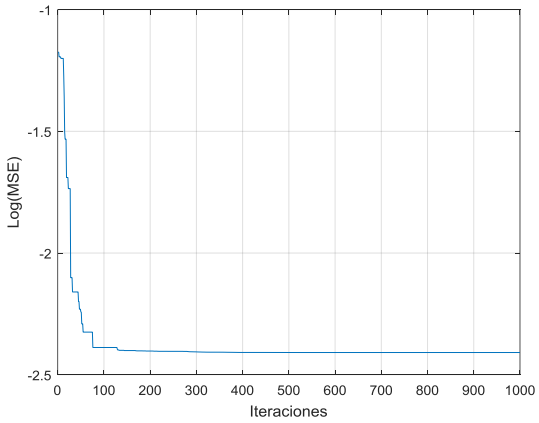

Figura 7: ANN (3,5,1), Fitness *MSE*, c_1=0.95, c_2=0. 95

Tabla 1: Resultados de la exactitud en la estimación, a partir de diferentes valores de c_1 y c_2

		Repetición 1		Repetición 2		Repetición 3		Promedio	
c_1	c_2	*RMSE*	R^2	*RMSE*	R^2	*RMSE*	R^2	*RMSE*	R^2
0.05	0.05	0.1937	53.0%	0.2209	38.9%	0.2088	45.4%	0.2078	45.8%
0.2	0.2	0.1859	56.8%	0.0830	91.7%	0.1595	74.5%	0.1428	74.3%
0.5	0.5	0.0640	94.9%	0.0656	94.6%	0.0671	94.5%	0.0655	94.7%
0.05	0.95	0.0615	95.3%	0.0671	94.8%	0.0607	95.4%	0.0631	95.2%
0.95	0.05	0.1031	88.2%	0.0985	89.0%	0.0641	95.0%	0.0885	90.7%
0.95	0.95	0.0623	95.1%	0.0611	95.3%	0.0610	95.3%	0.0614	95.2%
2.95	2.95	0.0618	95.3%	0.0611	95.4%	0.0635	94.9%	0.0621	95.2%
5.95	2.95	0.0612	95.2%	0.0618	95.2%	0.0626	95.3%	0.0618	95.2%
5.95	5.95	0.0612	95.3%	0.0773	95.0%	0.0655	95.1%	0.0680	95.1%
10	10	0.0675	95.3%	0.0656	95.1%	0.0626	95.2%	0.0652	95.2%

4 Conclusiones

PSO se puede definir como un método de optimización rápido, fácil de implementar y eficaz, en el cual los parámetros a sintonizar incluyen el peso inercial, las constantes de aceleración c_1 y c_2, el tamaño de la población y el límite superior de la velocidad v_{max}; sin entrar a valorar aún la influencia del tipo de topología de la población, la importancia determinante de la función de fitness o la inserción de técnicas alternativas para mitigar el riesgo de convergencia prematura inherente a PSO. En definitiva, la formulación de PSO se reduce a caracterizar el movimiento de las partículas en base a un operador velocidad que debe aunar exploración y convergencia, descomponiendo para ello la velocidad en tres componentes en un intento por sintetizar un comportamiento social.

En este contexto los valores de los coeficientes de aceleración de PSO c_1 y c_2 juegan un papel importante dentro del modelo del algoritmo: $c_1 > 0$ y $c_2 = 0$, las

partículas serán independientes; $c_1 = 0$ y $c_2 > 0$, entonces las partículas serán colectivas; $c_1 = c_2$, las partículas serán atraídas por un valor promedio; $c_1 > c_2$, la experiencia propia es mayor que la del grupo; $c_1 < c_2$, la experiencia del grupo es mayor que la propia; si c_1 y c_2 disminuyen, las trayectorias de desplazamiento de las partículas son suaves; y si, c_1 y c_2 aumentan, entonces los movimientos de las partículas serán abruptos.

La capacidad de ajustar las posiciones de las partículas en un espacio de soluciones satisfactoria, tomando en cuenta la función objetivo a minimizar (Error Cuadrático Medio - *MSE*) garantizó el éxito del algoritmo del experimento.

Dentro de la fase experimental, para valores muy pequeños de c_1 y c_2 (cercanos a cero) se obtiene una baja exactitud en la estimación de costos de fabricación de tubería. Una ANN ofrece mayor exactitud con coeficientes de aceleración mayores o iguales a 0.5, como lo indican en su estudio Duarte y Quiroga (2010).

Los resultados alcanzados en las métricas *RMSE* R^2 en este experimento utilizando una Red Neuronal Artificial tipo *feedforward* sigmoidal con aprendizaje PSO no difieren significativamente de los obtenidos por medio de una ANN recurrente aplicada por Barba y Bodero (2017), para analizar los mismos datos.

Conflicto de Intereses

Los autores declaran que no existe ningún tipo de Conflicto de Interés.

Referencias

Barba, L. & Rodríguez, N. (2015). Traffic Accidents Forecasting using Singular Value Decomposition and an Autoregressive Neural Network Based on PSO. *Polibits*, *51*, 33–38.

Barba, L. & Bodero, E. (2017). Redes Neuronales Artificiales para Estimación de Costos de Construcción Industrial. *V Congreso Internacional de Investigación y Actualización en Ingenierías*, *Galápagos (Ecuador),* 269-278.

Carlisle, A. & Dozier, G. (2001). An off-the-self PSO. *Proceedings of the Workshop on Particle Swarm Optimization*, Indianapolis (USA), 1-6.

Chang, R. & Lu, C.N. (2002). Feeder reconfiguration for load factor improvement. *Proceedings of the IEEE Power Engineering Society Winter Meeting*, New York (USA), Vol. 2, 980-984.

Duarte, C. & Quiroga, J. (2010). Algoritmo PSO para identificación de parámetros en un motor DC". *Revista Facultad de Ingeniería*, *Universidad de Antioquia*, *55*.

Eberhart, R. & Hu, X. (1999). Human tremor analysis using particle swarm optimization. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99*, Washington (USA), Vol. 3, 1927-1930.

Eberhart, R. & Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul (South Korea), Vol. 1, 81-86.

Franken, N. & Engelbrecht, A. P. (2004). PSO approaches to co-evolve IPD strategies. *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, Oregon (USA), 356-363.

Freeman, J. A. & Skapura, D. M. (1991). *Algorithms, Applications, and Programming Techniques*. USA: Addison-Wesley Publishing Company.

Gaing, Z. L. (2003). Discrete particle swarm optimization algorithm for unit commitment. *Proceedings of the 2003 IEEE Power Engineering Society General Meeting*, Toronto (Canadá), Vol. 1, 418-424.

Hu, X. & Eberhart, R. (2002). Adaptive particle swarm optimization: detection and response to dynamic systems. *Proceedings of the 2002 Congress on Evolutionary Computation-CEC02*, Honolulu (USA), Vol. 2, 1666-1670.

Hu, X., Eberhart, R. & Shi, Y. (2003). Engineering optimization with particle swarm. *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis (USA), 53-57.

Ismail, A. & Engelbrecht, A. (2000). Global optimization algorithms for training product unit neural networks. *Proceedings of IEEE-INNS-ENNS International Joint Conference on Neural Networks*, Como (Italy), Vol. 1, 132-137.

Kennedy, J. (1997). The particle swarm: social adaptation of knowledge. *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation-ICEC97*, Indianapolis (USA), 303-308.

Kennedy, J. & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of the IEEE International Conference on Neural Networks-ICNN'95*, Perth (Australia), Vol.4, 1942-1948.

Kennedy, J. & Eberhart, R. (2001). *Swarm intelligence*. San Francisco: Morgan Kaufmann Publishers.

Koay, C.A. & Srinivasan, D. (2003). Particle swarm optimization-based approach for generator maintenance scheduling. *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis (USA), 167-173.

Laskari, E., Parsopoulos, K. & Vrahatis, M. (2002). Particle swarm optimization for minimax problems. *Proceedings of the 2002 Congress on Evolutionary Computation-CEC02*, Honolulu (USA), Vol. 2, 1576-1581.

Lu, Z-S. & Yan, S. (2004). Multiuser detector based on particle swarm algorithm. *Proceedings of the IEEE 6th CAS Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication*, Shanghai (China), 783-786.

Millonas, M.M. (1994). Swarms, phase transitions, and collective intelligence. *Proceedings of Artificial life III*, Vol. XVII, SFI Studies in the Sciences of Complexity, Addison-Wesley.

Miranda, V. & Fonseca, N. (2002). EPSO – Evolutionary particle swarm optimization, a new algorithm with applications in power systems. *IEEE/PES Transmission and Distribution Conference and Exhibition*, Porto Alegre (Brazil), Vol. 2, 745-750.

Naka, S., Genji, T., Yura, T. & Fukuyama, Y. (2001). Practical distribution state estimation using hybrid particle swarm optimization. *Proceedings of 2001 Winter Meeting of the IEEE Power Engineering Society*, Columbus (USA), Vol. 2, 815-820.

Naka, S., Genji, T., Yura, T. & Fukuyama, Y. (2003). A hybrid particle swarm optimization for distribution state estimation. *IEEE Transactions on Power Systems*, Vol. 18, No. 1, 60-68.

Rodriguez, N. & Duran, O. (2013). Reduced Multivariate Polynomial Model for Manufacturing Costs Estimation of Piping Elements. *Mathematical Problems in Engineering*, Vol. 2013.

Srinivasan, D., Loo, W. & Cheu, E. (2003). Traffic incident detection using particle swarm optimization. *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis (USA), 144-151.

Tasgetiren, M.F. & Liang, Y-C. (2003). A binary particle swarm optimization algorithm for lot sizing problem. *Journal of Economic and Social Research*, Vol. 5, No. 2, 1-20.

Veeramachaneni, K. & Osadciw, L.A. (2004a). Optimal scheduling in sensor networks using swarm intelligence. *Proceedings of the Conference on Information Sciences and System*, Princeton University, New Jersey (USA).

Veeramachaneni, K. & Osadciw, L.A. (2004b). Dynamic sensor management using multi objective particle swam optimizer. *Proceedings of the SPIE*, Vol. 5434, Multisensor, Multisource Information Fusion: Architecture, Algorithms, and Applications 2004, 205-216.

Vesterstrom, J. & Riget, J. (2002). Particle swarms: Extensions for improved local, multi-modal, and dynamic search in numerical optimization (Master's Thesis). University of Aarhus.

Wang, Z., Durst, G., Eberhart, R., Boyd, D. & Miled, Z. (2004). Particle swarm optimization and neural network application for QSAR. *Proceedings of the 18th International Parallel and Distributed Processing Symposium* (IPDPS'04), New Mexico, (USA).

Yangyang, Z., Chunlin, J.I., Ping, Y., Manlin, L.I., Chaojin, W. & Guangxing, W. (2004). Particle swarm optimization for base station placement in mobile communication. *Proceedings of the 2004 IEEE International Conference on Networking, Sensing & Control*, Taipei (Taiwan), 428-432.

Zhao, Y. & Zheng, J. (2004). Particle swarm optimization algorithm in signal detection and blind extraction. *Proceedings of the 7th International Symposium on Parallel Architectures*, Hong Kong (China), 37-41.